An Equation-Based Approach to Transition Specifications in Material Flow Networks

Andreas Moeller[1]

Abstract

Material flow analysis (MFA) describes a class of methods that allow analysing and designing material and energy flow systems like supply chains, eco-industrial parks or production sites. Material flow networks are one of the methods in the field of MFA. Based on the Petri nets, the material flow network make it possible to conduct static MFA (steady-state modelling) as well as dynamic MFA (continuous and discrete-event simulation). In this contribution focus will be on the important building blocks of material flow networks: transitions. Transition specifications can be regarded as parameterized sub-models, which may exhibit nonlinear behaviour. Linear transition specifications that consist of fixed production coefficients are always not adequate to represent the sub-models. In the following a more powerful equation-based approach to process flowsheeting is discussed. The equation-based approach makes it possible to specify a transition with aid of a system of non-linear algebraic equation including parameters, design specifications and loops.

1. Introduction

Material flow networks as a modelling framework provide different means to specify unit processes that represent material and energy transformations. In the Petri net terminology these processes are called transitions. In software systems like Umberto®, which are based on material flow networks, the default overall solver implements a sequential modular strategy to calculate all material and energy flows, stocks and other performance indicators of a given material flow model.

The solver considers the transition specifications as sub-models with their own solvers. These solvers of the sub-models are called to calculate all input and output flows as well as other performance indictors of a single transition. Such a modular approach provides a very flexible and simple interface to sub-models:

1) Specifications with aid of production coefficients (‘Linear Specifications’): This kind of specification is common in life cycle assessment (non-linear specifications are not allowed) so that libraries like EcoInvent support this kind of specification.
2) Much more flexible are specifications that allow using user-defined expressions (‘User-Defined Functions’). The expressions are assigned to identifiers that represent input and output flows. The whole specification consists of a set of assignments. Non-linear expressions are possible.
3) The most flexible approach is to provide a solver that allows specifying the solution with aid of programming languages like Python (‘Scripting’).

Other solvers are reasonable, for instance an Excel-based approach (‘Spread sheet mind-set’). In this contribution, a method is presented that is very common in chemical engineering: a system of non-linear algebraic equations and variables specify a transition. Such an equation-based approach is similar to ‘user-defined functions’ but more powerful and flexible. Moreover, it allows applying equation-based overall solvers, similar to equation-based strategies for process flowsheeting. On the other side, an equation-based approach is not always in line with our sequential mental models so that it is not easy to specify a transition in such a way. For instance, the degree of freedom for

[1] Leuphana University Lueneburg, Germany, moeller@uni.leuphana.de, Institute for Sustainability Communication

the equation system must be zero. Special algorithms are required to identify fully determined, over-determined and under-determined parts of the system.

2. Approaches to Process Flowsheeting

Process flowsheeting is a computer-based design process in chemical engineering. “As used here the term ‘flowsheeting’ is the use of computer aids to perform steady-state heat and material balancing, and sizing and costing calculations for a chemical process” [1]. In fact, the application context and the engineering problems are very similar to material flow analysis (MFA) so that the approaches and algorithms can be applied here too.

The purpose of software tools is to support modelling experts to construct the models, to validate them and to calculate results. The most important category of results in the field of MFA is the category of material and energy flows. For a given material and energy flow model, which consists of a flow diagram or net, process or transition specifications, already known flows and stocks and design or scenario parameters, the software tool calculates all material and energy flows in a consistent way.

Data sources of the calculation procedures are transition specifications. Transition specifications are sub-models that allow calculating the input and output flows for a single transition. As mentioned above, software tools can implement different approaches. From a computer science perspective, the sub-model is an algorithm that determines all inputs and outputs for a given sequence of input parameters. In engineering, it is very common to specify a sub-model by a set of model equations.

Software tools to process flowsheeting implement different design images, for instance the sequential modular approach (or ‘sequential modular strategy’) or the equation-based approach (or ‘equation oriented strategy’) [1, 2, 3]. The sequential modular approach regards the sub-models as black-boxes that implement software interfaces. The solver uses the interface implementations to calculate material and energy flows. “There are subroutines for the flash unit, distillation columns, absorbers, a variety of reactor types, compressors, pumps, valves, and so forth. One constructs a complete process model by wiring up an appropriate set of these building blocks. The flowsheeting system then solves the total process model by calling each of the unit models in turn, according to how they are wired together, iterating where necessary to converge complex process models” [3]. The solver starts with given ‘feed streams’ or ‘manual flows’. Then the solver checks the transition specifications. If all input parameters are already assigned and fixed, the subroutine will be called. As a result of such a subroutine call, more flows are calculated. The resulting main loop is iterated as long as all material and energy flows are calculated or no more subroutines can be called For example, the default solver of the MFA tool Umberto® implements the sequential modular approach [4].

The main problem of sequential modular strategies is the handling of loops (recycling). The major approach to deal with loops includes flowsheet ordering, tearing and the application of convergence methods [1, 3]. An advantage of the sequential modular approach is that the calculation procedure is in line with our ‘sequential mind-set’. This makes model validation easier. Another advantage is that it is a modular approach. It is not necessary that the overall solvers know how the calculation of single processes or transitions is implemented. This makes it easier to realize a flexible plugin-in based software system. The main disadvantage is that the modelling experts construct not only the material and energy model, the sequential modular approach normally requires manual support, in particular manual tearing and initial guesses [5]. The model specification is not only declarative but also procedural [3].

Software systems that implement the equation-based approach expect that single processes or transition specification provide a set of nonlinear algebraic equations as model equations, based on a overall grammar (specification language), so that the solver can evaluate the equations: collection of variables used in the equations, calculation and maybe symbolic differentiation to obtain the derivatives [6]. So, "the final flowsheet is represented by a collection of nonlinear equations which must be solved simultaneously. The equations include the model equations and connection equations" [1], and "additional specifications are added until the degrees of freedom for the equation system are zero and a well posed mathematical problem remains" [2], for instance manual flows, scenario parameters and material properties.

To solve a system of nonlinear algebraic equations [7], the compiled system of equations is transformed into a root finding problem (the left side of all equation contains an algebraic expression, the right side zero). The left sides of all equations is regarded as a multi-dimensional function f, and the purpose of multi-dimensional root finding algorithms is to find an assignment to all variables (vector x) so that f(x)=0. In other words, "the heart of any equation oriented simulator is the multidimensional root finding code" [2]. However, to reduce the number of equations, a preparatory step seems to be beneficial. Such a step divides the system of nonlinear equations into a sequence of components. The solver determines all variables of the system by calculating the components in a sequential manner so that the root finding algorithm must be applied to the components and not to the whole system. These calculation steps and appropriate algorithms are described in the following.

3. Equation-based Approach

An overall solver that implements the sequential modular approach integrates two levels of calculation: the calculation of single processes or transitions and the calculation of flowsheets or networks. "In fact, the sequential modular approach can be best interpreted as a decomposition approach in which a two level nested iteration is established. The outer iteration uses a tearing approach to converge, and the inner iteration uses the unit module specific methods to converge. Further, the variables in the outer iteration (the tear variables) are chosen so that the inner iteration breaks down into a series of subproblems (corresponding to the unit modules) that can be solved sequentially" [2]. In other words, the sequential modular approach as an overall strategy can be combined with the equation-based approach on the level of single transitions. In the following, such a combination of the two approaches is introduced. As a side effect, the number of nonlinear equations is limited and does not incorporate all life cycle stages from raw material extraction to waste disposal.

As mentioned above, a solver for single processes or transitions that implements the equation-based approach consists of two different steps: Block decomposition as a preparatory step and root finding for each component.

3.1. Block decomposition

The purpose of the block decomposition step is to divide the whole system of algebraic equations into a sequence of strongly connected components. The strongly connected components are "those minimal subsets of units that must be solved simultaneously" [2]. These components constitute again systems of nonlinear equations but of lower degree. The sequence of strongly connected components can be solved step-by-step. Block decomposition includes the validation of the system.

Block decomposition consists of two steps [2, 8]. The first step constructs a directed graph that represents the dependencies between the equations. Therefore, all not already fixed variables of all equations are determined. Based on the collections of variables for each equation, an occurrence matrix m is derived, one column per variable and one row per equation. If a variable j occurs in

equation i, the value of m(i,j) is 1 otherwise 0. With aid of Duff's algorithm, that calculates a maximum transversal of a matrix [9], the digraph representation of the system can be constructed: As a result of Duff's algorithm one variable can be assigned to each equation, and the edges of the directed graph represent the dependency from other variables or equations respectively. Thereafter, Tarjan's algorithm is applied to obtain the strongly connected components and a precedence order is applied [2, 10, 11]. The root finding problem is reduced to each of the strongly connected components.

3.2. Root Finding

Root finding is the core component of any solver for systems of non-linear equations. Therefor, all equations of a given strongly connected component $g_i(x) = h_i(x)$ are transformed to $g_i(x) - h_i(x) = 0$ (where i is the index of the equation, n the number of not already fixed variables and x the vector of not already fixed variables). This yields the multi-dimensional function

$$f: R^n \rightarrow R^n \tag{1}$$

with the components $f_i := g_i - h_i$. A solution of the system of nonlinear equations can be determined by finding the roots of f (1). Here, mainly the multidimensional Newton's method or a Quasi-Newton method like Broyden is applied [1, 2, 3, 6, 12].

Starting with an estimation x_0 for all variables, Newton's method results in an iterative procedure

$$x_{i+1} = x_i - f(x_i)(J(x_i))^{-1} \tag{2}$$

where $J(x_i)$ is the Jacobian matrix of all partial derivatives of *f*. If possible, the derivatives can be obtained by symbolic differentiation. An alternative is to use finite differences as an approximation [6]. The second alternative is more flexible because this allows user-defined functions, implemented e.g. with aid of scripting languages like Python. But this could be a time consuming task. Another time-consuming procedure is to obtain the inverse of $J(x_i)$.

Broyden's method tries to avoid the determination of the Jacobian matrix in all iterations [13, 14]. Instead, so-called update formulas are used to derive the Jacobian (or the inverse) of iteration i+1 from the Jacobian of iteration i.

Another challenge is to estimate the starting vector x_0. It can happen that it is not possible to determine a root with aid of Quasi-Newton methods even if a root exists. This depends mainly on a good first estimation. However, in particular in material flow analysis the model equations exhibit normally a non-problematic behaviour; many process specifications are linear. In such a case, root finding may start with values that occur normally in the material and energy flow model (for instance typical flow values). Of course, modelling experts need the opportunity to specify first estimations.

4. Integration

The different options to specify transitions have their advantages and disadvantages. For instance, linear specifications are normally the starting point and easy to understand whereas Python scripts require knowledge in software programming. To specify a transition as simple as possible, it is reasonable to combine the different options within a single transition specification, for instance the combination of the equation-based approach with linear production coefficients (emission factors for hundreds of emissions) or the use of Python functions in equations. This can be considered as 'manual block decomposition': The whole specification consists of different section, for instance sections to specify Python functions, direct assignments, equations or linear coefficients. The solver calculates the sections step-by-step. Finally, all variables should be calculated in a consistent manner.

5. Example

The 'Ammonia Process' described in [15] is used to illustrates the application domain of the new type of transition specification. The flowsheet model (figure 1) consists of three unit processes: mixer, chemical reactor and a separator unit. The separator unit is necessary because the conversation rate of the chemical reaction is assumed to be about 25%.

$$1N_2 + 3H_2 \leftrightarrow 2NH_3$$

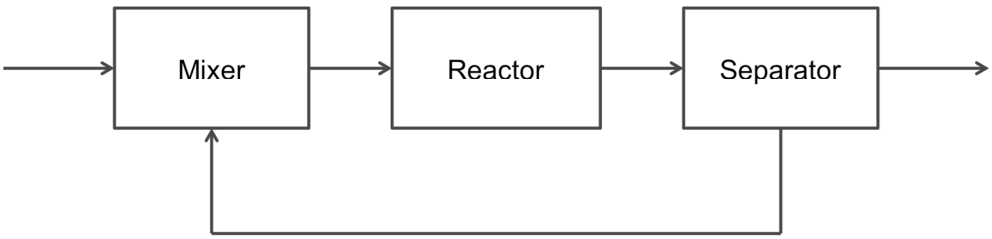

Figure 1: Flowsheet of an ammonia process [15]

The feed stream on the input side of the mixer consists of 100 kMol N_2 (nitrogen) and 300 kMol H_2 (hydrogen). This example is easy to validate because we can expect about 200 kMol ammonia as an output stream (in steady state the overall conversion rate is almost 100%). However, the separator unit cannot remove all nitrogen and hydrogen from the output stream.

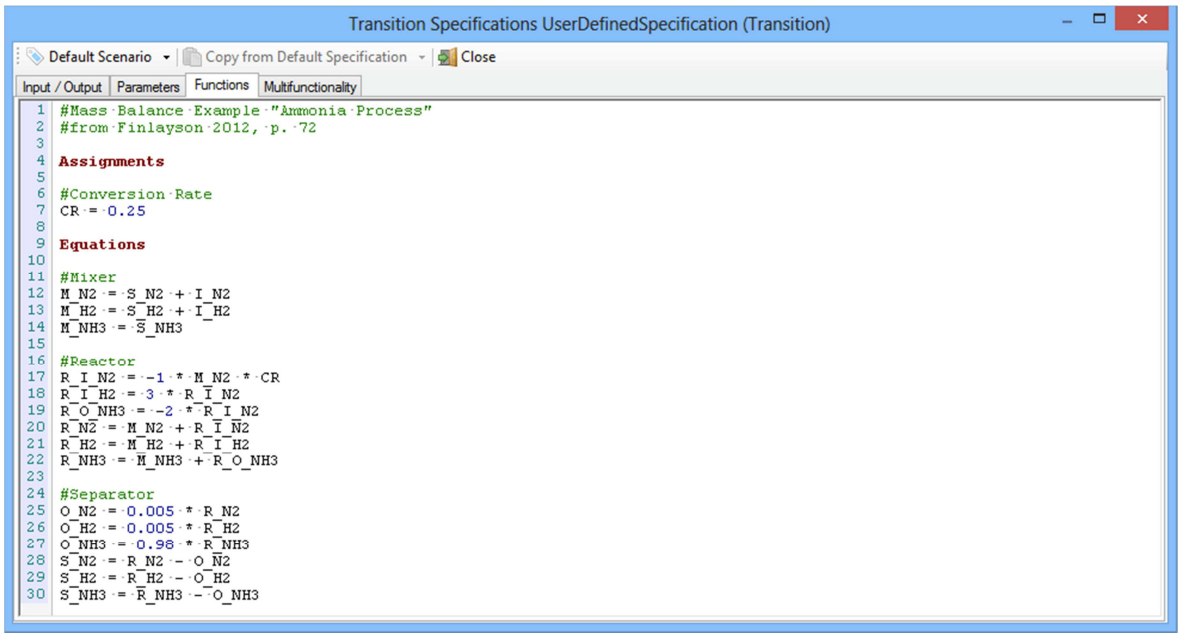

Figure 2: Transition specification of the ammonia process [15]

Figure 2 shows the resulting transition specification. It consists of two sections, the specification of the conversion rate and equations for the mixer, chemical reactor and the separator unit. The variables of input flows (fixed feed streams N2 and H2) use the prefix I_ and the output flows O_. The internal variables use the first character of the unit process that produces them, for instance M_N2, M_H2 and M_NH3 are the output streams of the mixer process.

An application of a sequential procedure to calculate all variables is not possible. In particular the variables S_N2 and S_H2 occur in line 12 or 13 and again in 28 or 29.

The calculation engine of the modelling tool analyses the system of equations, determines the collection of variables for each equation and constructs the occurrence matrix. The following table shows the first row and columns of the occurrence matrix (all equations of the mixer). The variables I_N2 and I_H2 do not appear in the matrix because they are already fixed variables (I_N2 = 100000 and I_H2 = 300000).

	S_N2	M_N2	S_H2	M_H2	S_NH3	M_NH3	…
S_N2 + I_N2 – (M_N2)	1	1	0	0	0	0	…
S_H2 + I_H2 – (M_H2)	0	0	1	1	0	0	…
S_NH3 – (M_NH3)	0	0	0	0	1	1	…
…	…	…	…	…	…	…	…

The block decomposition step yields the following strongly connected components. It is interesting to understand how Tarjan's algorithm has divided the system into strongly connected components. The first component can be called the 'nitrogen', the third 'hydrogen' and the last 'ammonia', with two 'connecting' components between them:

```
Strongly connected component 1:
0.005 * R_N2 – (O_N2)            # assigned variable = O_N2
-1 * M_N2 * CR - (R_I_N2)        # assigned variable = R_I_N2
 S_N2 + I_N2 - (M_N2)            # assigned variable = M_N2
 M_N2 + R_I_N2 - (R_N2)          # assigned variable = R_N2
 R_N2 - O_N2 - (S_N2)            # assigned variable = S_N2

 Strongly connected component 2:
 3 * R_I_N2 - (R_I_H2)           # assigned variable = R_I_H2

 Strongly connected component 3:
 R_H2 - O_H2 - (S_H2)            # assigned variable = O_H2
 0.005 * R_H2 - (O_H2)           # assigned variable = R_H2
 M_H2 + R_I_H2 - (R_H2)          # assigned variable = M_H2
 S_H2 + I_H2 - (M_H2)            # assigned variable = S_H2

 Strongly connected component 4:
 -2 * R_I_N2 - (R_O_NH3)         # assigned variable = R_O_NH3

 Strongly connected component 5:
 R_NH3 - O_NH3 - (S_NH3)         # assigned variable = O_NH3
 0.98 * R_NH3 - (O_NH3)          # assigned variable = R_NH3
 M_NH3 + R_O_NH3 - (R_NH3)       # assigned variable = M_NH3
 S_NH3 - (M_NH3)                 # assigned variable = S_NH3
```

Thereafter, Broyden's root finding method is applied to each strongly connected component. Because no first estimations are specified, all coefficients of the starting vector x_0 are 1. The following figure shows the calculation log. The algorithm needs 3 iterations to determine the variables for the most strongly connected components. This includes O_NH3, O_N2 and O_H2 but internal variable may be important performance indicators too. For instance, the sum of R_NH3, R_H2 and R_N2 could serve as a performance indicator to estimate the energy consumption.

The example shows a typical application domain of the new type of transition specification: chemical processes as sub-models in material flow networks. The basic idea behind is to use the material flow networks as a modelling approach on a higher level and to combine the material flow

networks with process flowsheeting on the level of single transitions. The flowsheet models become sub-models in such a modelling environment.

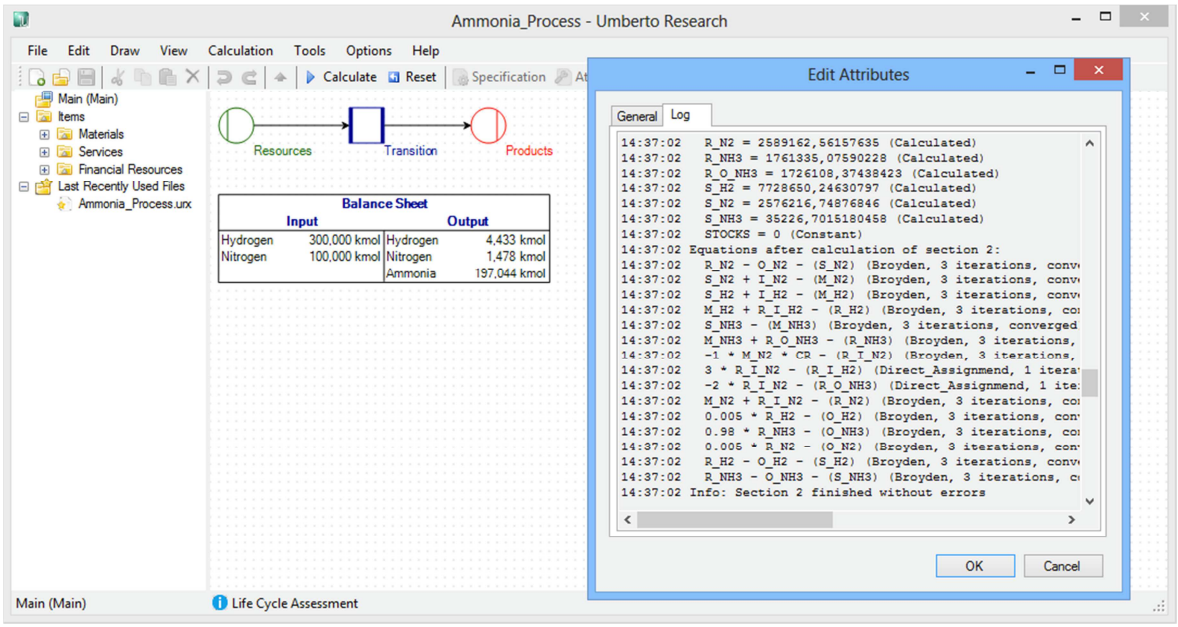

Figure 3: Input/output balance and calculation log of the software tool

A next implementation step could be to develop an appropriate user interface so that users can construct the flowsheet with aid of a graphical user interface. Nevertheless, such an extension of MFN tools cannot replace specialized process flowsheeting systems.

6. Outlook

The extension of transition specifications options combines the most important approaches to steady-state modelling: the sequential modular approach on the level of whole material flow networks and the equation-based approach on the level of single transitions. It seems to be attractive apply the equation-based approach to whole material flow networks as an alternative to the sequential modular strategy. However, this works fine for small networks, which consist of user-defined transitions and equation-based transition specifications as described above. So, it is likely to implement this on the level of sub-nets.

One of the problems is that the interface between the overall calculation engine and the transition specifications is different. Modular approaches expect that the transition specifications include a solver. This makes it possible to implement a flexible and extensible interface, including Excel® or process flowsheeting tools as a transition specification plugins. Another option is to implement a wrapper to computer-based corporate information systems, in particular ERP systems. A solution to this problem could be to implement an approach that tries to combine the advantages of the sequential modular approach and the equation-based approach: the simultaneous modular strategy [1, 5].

A second challenge is that the material flow networks support dynamic MFA. Here, emphasis is on the development and future availability of stocks. Before applying the equation-based and the simultaneous approach to whole material flow networks, it is necessary to combine the approaches to steady state modelling with approaches in the field of continuous simulation. This results in a two-level calculation engine: steady-state approaches for the inner loop, integration methods for the outer loop.

References

[1] Westerberg, A.W., Hutchinson, H.P., Motard, R.L. and Winter, P., *Process flowsheeting*. Cambridge, London, New York, Melbourne: Cambridge University Press, 1979.

[2] Barton, P. I., *The Equation Oriented Strategy for Process Flowsheeting*, Department of Chemical Engineering, Massachusetts Institute of Technology, Cambridge, MA, 2000.

[3] Westerberg, A.W. and Piela, P.C., *Equational-based process modeling, technical report*, Department of Chemical Engineering, Carnegie Mellon University, Pittsburgh, 1994.

[4] Moeller, A., Grundlagen stoffstrombasierter Betrieblicher Umweltinformationssysteme, Bochum: Projekt Verlag, 2000.

[5] Moeller, A., "Applications of the Simultaneous Modular Approach in the Field of Material Flow Analysis," 2013. In *Environmental Informatics and Renewable Energies: 27th International Conference on Informatics for Environmental Protection*, ed. Page, B., Fleischer, A., J. G. & Wohlgemuth, V., 456-464. Aachen: Shaker Verlag.

[6] Biegler, L.T., *Systems of Nonlinear Equations – Part II*, Chemical Engineering Department, Carnegie Mellon University, Pittsburgh, 2000.

[7] Chen, H.-S. & Stadtherr, A., "On Solving Large Sparse Nonlinear Equation Systems," *Computers & Chemical Engineering* vol. 8, no. 1, pp. 1-7, 1984.

[8] Frenkel, J., Kunze, G. & Fritzson, P.: "Survey of appropriate matching algorithms for large scale systems of differential algebraic equations," In *2012 Proceedings of the 9th International Modelica Conference*, pp. 433-442.

[9] Duff, I.S., "On algorithms for obtaining a maximum transversal," *ACM Transactions of Mathematical Software* vol. 7, pp. 315–330, 1981.

[10] Duff, I.S., Reid, J.K., "An Implementation of Tarjan's Algorithm for the Block Triangularization of a Matrix," *ACM Transactions on Mathematical Software* vol. 4, no. 2, pp. 137-147, 1978.

[11] Tarjan, R.E., "Depth-first search and linear graph algorithms," *SIAM Journal on Computing* I, pp. 146-160, 1972.

[12] Broyden, C.G., "A Class of Methods for Solving Nonlinear Simultaneous Equations," *Mathematics of Computation* vol. 19, pp. 577-593, 1965.

[13] Burkard, R.E., Zimmermann, U.T., *Einführung in die Mathematische Optimierung*, Berlin, Heidelbarg, New York: Springer, 2012.

[14] Henning, P., Kiefel, M., "Quasi-Newton Methods: A New Direction," *Journal of Machine Learning Research* vol. 14, pp.843-865, 2013.

[15] Finlayson, B.A., *Introduction to Chemical Engineering Computing*, 2nd ed., Hoboken, New Jersey: John Wiley & Sons, 2012.